Partitioning Sparse Matrices with Eigenvectors of Graphs

Alex Pothen[1], Horst D. Simon[2], and Kang-Pu Paul Liu

Report RNR-89-009, July 1989

NAS Systems Division
NASA Ames Research Center, Mail Stop T-045-1
Moffett Field, CA 94035

July 25, 1989

[1]The first and third author are with the Computer Science Department, The Pennsylvania State University, University Park PA 16802

[2]The second author is an employee of Boeing Computer Services, Bellevue, WA 98128

PARTITIONING SPARSE MATRICES WITH EIGENVECTORS OF GRAPHS

ALEX POTHEN*, HORST SIMON† AND KANG-PU PAUL LIOU‡

Abstract. The problem of computing a small vertex separator in a graph arises in the context of computing a good ordering for the parallel factorization of sparse, symmetric matrices. An algebraic approach to computing vertex separators is considered in this paper. It is shown that lower bounds on separator sizes can be obtained in terms of the eigenvalues of the Laplacian matrix associated with a graph. The Laplacian eigenvectors of grid graphs can be computed from Kronecker products involving the eigenvectors of path graphs, and these eigenvectors can be used to compute good separators in grid graphs. A heuristic algorithm is designed to compute a vertex separator in a general graph by first computing an edge separator in the graph from an eigenvector of the Laplacian matrix, and then using a maximum matching in a subgraph to compute the vertex separator. Results on the quality of the separators computed by the spectral algorithm are presented, and these are compared with separators obtained from Automatic Nested Dissection and the Kernighan-Lin algorithm. Finally, we report the time required to compute the Laplacian eigenvector, and consider the accuracy with which the eigenvector must be computed to obtain good separators. The spectral algorithm has the advantage that it can be implemented on a medium size multiprocessor in a straight forward manner.

AMS(MOS) subject classifications: 65F50, 65F05, 65F15, 68R10.

Keywords. graph partitioning, graph spectra, Laplacian matrix, sparse matrix, vertex separator, parallel reordering algorithms.

1. Introduction.

In the solution of large, sparse, positive definite systems on parallel computers, it is necessary to compute an ordering of the matrix such that it can be factored efficiently in parallel. For large problems, the storage required for the structure of the matrix may exceed the storage capacities of a single processor, and this parallel ordering itself will need to be computed in parallel. One strategy to compute a good parallel order is to employ the divide and conquer paradigm: Find a set of vertices in the adjacency graph of the matrix, whose removal disconnects the graph into two nearly equal parts. Number the vertices in the separator last, and recursively number the vertices in the two parts by the same strategy. This strategy is employed in the well-known Nested Dissection algorithm for ordering matrices for sparse factorization.

In computing an ordering by the above approach, at each step, the following *Partitioning problem* needs to be solved. Given an adjacency graph G of a sparse matrix, this problem is to find a vertex separator S such that S has few vertices and S disconnects $G \setminus S$ into two parts A, B, with nearly equal numbers of vertices.

* Computer Science Department, The Pennsylvania State University, Whitmore Lab, University Park, PA 16802. Electronic address: pothen@shire.sys.cs.psu.edu, na.pothen@na-net.stanford.edu. The research of this author was supported by National Science Foundation Grant CCR-8701723 and U.S. Air Force Office of Scientific Research Grant AFOSR-88-0161.

† Numerical Aerodynamic Simulation (NAS) Systems Division, Mail-Stop 258-5, NASA Ames Research Center, Moffett Field, CA 94035. The author is an employee of Boeing Computer Services. Electronic address: simon@orville.nas.nasa.gov

‡ Computer Science Department, The Pennsylvania State University, Whitmore Lab, University Park, PA 16802.

In this paper, we consider a spectral algorithm for solving the partitioning problem. We associate with the given sparse, symmetric matrix (and its adjacency graph), a matrix called the Laplacian matrix. We compute a particular eigenvector of the Laplacian matrix, and use its components to initially partition the vertices into two sets A', B'. The set of edges joining A' and B' is an edge separator in the graph G. A vertex separator S is computed from the edge separator by a matching technique.

This spectral algorithm has three features that distinguish it from previous algorithms that are worthy of comment.

First, previous algorithms for computing separators, such as Nested Dissection or the Kernighan-Lin algorithm make use of *local* information in the graph, viz. information about the neighbors of a vertex, to compute separators. The spectral method employs *global* information about the graph, since it computes a separator from eigenvector components. Thus the spectral method has the potential of finding separators in the graph that are qualitatively different from the separators obtained by previous approaches.

Second, we can view the spectral method as an approach in which a vertex in the graph makes a *continuous* choice, with a weight between $+1$ and -1, about which part in the initial partition it is going to belong to. All vertices with weights below the median weight form one part, and the rest, the other part. In the Kernighan-Lin method, each vertex makes a discrete choice (zero or one) to belong to one set. The weights in the spectral method can be used to move a few vertices from one part to the other, if a slightly different partition is desired in the course of the separator algorithm.

Third, the dominant computation in the spectral method is an eigenvector computation by a Lanczos or similar algorithm. This distinguishes the new algorithm from standard graph theoretical algorithms computationally. Most of the computation is based on standard vector operations on floating point numbers. Becasue of its algebraic nature the algorithm is parallelizable in a fairly straight forward manner on medium-grain multiprocessors used in scientific computing. Furthermore, since most of the computations are also vector floating point operations, this algorithm is well suited for vector supercomputers used for large scale scientific computing.

This paper is organized as follows. We include background material on the spectral properties of Laplacian matrices and their relevance to graph partitioning in § 2. We also review earlier work on computing edge separators from the eigenvectors of the adjacency matrix in this section. In § 3, we obtain lower bounds on the size of the best vertex separators of a graph in terms of the eigenvalues of the Laplacian matrix. Two different techniques for proving lower bounds are illustrated: One uses the Courant-Fischer minimax criterion, and the second employs an inequality from the proof of the Wielandt-Hoffman theorem. We then show that the spectra of rectangular and square grid graphs can be computed explicitly from the spectra of path graphs by employing suitable graph products and Kronecker products in § 4. We proceed to show how good edge and vertex separators in the grid graphs can be computed from the spectral information. In § 5, we describe our heuristic spectral algorithm to compute vertex separators in general graphs. The algorithm initially computes an edge separator, and then uses a maximum matching in a subgraph to compute the vertex separator. Results about the quality of the separators computed by the algorithm is presented

in § 6. In this section, we also compare the spectral separators with separators computed by Automatic Nested Dissection and the Kernighan-Lin algorithm, as well as with results obtained recently by Liu [31] and Leiserson and Lewis [29]. The time required to compute the Laplacian eigenvectors with the Lanczos algorithm and the accuracy needed in the eigenvector to obtain good separators are addressed in § 7. The final § 8 contains our conclusions and some directions for future work.

2. Background. Let $G = (V, E)$ be an undirected graph on $|V| = n$ vertices. The $n \times n$ adjacency matrix $A = A(G)$ has element $a_{v,w}$ equal to one if $(v, w) \in E$, and zero otherwise. By convention, $a_{v,v}$ is zero, for all $v \in V$. The rows and columns of the matrices associated with a graph are indexed by the vertices of the graph, their order being arbitrary. Let $d(v)$ denote the degree of a matrix, and define D to be the $n \times n$ diagonal matrix with $d_{v,v} = d(v)$. The matrix $Q = Q(G) = D - A$ is the *Laplacian matrix* of G.

Let the edges of the graph G be directed arbitrarily, and let C denote the vertex-edge incidence matrix of the directed graph. The $|V| \times |E|$ matrix C has elements

$$c_{v,e} = \begin{cases} +1 & \text{if } v \text{ is the head of } e \\ -1 & \text{if } v \text{ is the tail of } e \\ 0 & \text{otherwise.} \end{cases}$$

It is easy to verify [7] that $Q(G) = CC^t$, and that Q is independent of the direction of the edges in C.

The spectral properties of Q have been studied by several authors. Since

$$\begin{aligned} \underline{x}^t Q \underline{x} &= \underline{x} C C^t \underline{x} \\ &= (C^t \underline{x})^t (C^t \underline{x}) \\ &= \sum_{(v,w) \in E} (x_v - x_w)^2, \end{aligned}$$

it is easily seen that Q is positive semidefinite. Let the eigenvalues of Q be ordered

$$\lambda_1 = 0 \leq \lambda_2 \leq \ldots \lambda_n.$$

An eigenvector corresponding to λ_1 is $\underline{e}$, the vector of all ones. The multiplicity of the zero eigenvalue is equal to the number of connected components of the graph. If G is connected, then the second smallest eigenvalue λ_2 is positive. We call an eigenvector $\underline{y}$ corresponding to λ_2, a *second eigenvector.*

Fiedler [17, 18] has studied the properties of the second eigenvalue λ_2 and a corresponding eigenvector $\underline{y}$. He calls λ_2 the *algebraic connectivity*, and relates it to the vertex and edge connectivities of a graph. He has also investigated the partitions of G generated by the components of the eigenvector $\underline{y}$. One of his results of interest in this paper is the following.

THEOREM 1. *Let G be a connected graph, and let $\underline{y}$ be an eigenvector corresponding to λ_2. For a real number $r \geq 0$, define $V_1(r) = \{v \in V : y_v \geq -r\}$. Then the subgraph induced by $V_1(r)$ is connected. Similarly, for a real number $r \leq 0$, let the set $V_2(r) = \{v \in V : y_v \leq |r|\}$. The subgraph induced by $V_2(r)$ is also connected.*

In both sets V_1 and V_2, it is necessary to include all vertices with zero components for the theorem to hold. The role played by these latter vertices in the connectedness of the two subgraphs has been investigated at greater length by Powers [40, 41].

A corollary to this result is that if $y_v \neq 0$ for all $v \in V$, then the subgraphs induced by the sets

$$\begin{aligned} P &= \{v \in V : y_v > 0\}, \\ N &= \{v \in V : y_v < 0\} \end{aligned}$$

are both connected subgraphs of G.

The eigenvectors of the adjacency matrix corresponding to its algebraically largest eigenvalues have also been used to partition graphs. It is of interest to ask if a similar theorem holds for an eigenvector corresponding to the second largest eigenvalue of the adjacency matrix.

Let $\underline{x}$, $\underline{y}$ denote eigenvectors corresponding to the algebraically largest and second largest eigenvalues, respectively, of the adjaecncy matrix of G. By Perron-Frobenius theory, it is known that all components of $\underline{x}$ are positive. If α is a nonnegative number, then the subgraph induced by $V_1 = \{v \in V : y_v + \alpha x_v \geq 0\}$ is connected. Similarly, if α is a nonpositive number, then the subgraph induced by $V_2 = \{v \in V : y_v - |\alpha| x_v \leq 0\}$ is also connected.

Mohar [33] has made use of the second Laplacian eigenvector $\underline{y}$ to study the *isoperimetric number*, $i(G)$. If U is a subset of the vertices of the graph G, and δU denotes the set of edges with one endpoint in U and the other in $V \setminus U$, then

$$i(G) = \min_{|U| \leq n/2} \frac{|\delta U|}{|U|}.$$

Mohar proved that

$$\frac{1}{2}\lambda_2 \leq i(G) \leq (\lambda_2(2\Delta - \lambda_2))^{1/2},$$

where Δ is the maximum degree of a vertex. The proof of the upper bound makes use of the set $W = \{v \in V : y_v > 0\}$, with the sign of the eigenvector chosen such that W has no more than $n/2$ vertices.

The relationship of the Laplacian spectrum to several graph properties have been considered by several authors; two recent survey articles by Mohar [34] and Bien [6] describe some of these results.

Spectral methods for computing edge separators have been considered by several researchers.

Donath and Hoffman [12] obtain lower bounds on the size of an edge separator in terms of the eigenvalues of the matrix $A + U$, where A is the adjacency matrix, and U is a diagonal matrix with the property that $\text{trace}(U) = -2|E|$. Barnes and Hoffman [4] consider graphs with weights on the edges, and obtain lower bounds on the weights of edge separators in terms of the eigenvectors corresponding to the algebraically largest eigenvalues of a matrix $A + D$. Here, A is the weighted adjacency matrix, and D is an $n \times n$ matrix such that the elements of $A + D$ sum to zero. They formulate the partitioning problem as a quadratic integer

programming problem, and then approximate the latter problem by linear programming problems, whose solutions yield the lower bounds.

Barnes [3] shows that the problem of partitioning a graph into $k \geq 2$ parts such that the number of edges cut by the partition is minimum is equivalent to a matrix approximation problem, in which the adjacency matrix is approximated by a *partition matrix* P. The $n \times n$ matrix P has element $p_{i,j}$ equal to one if vertices i and j belong to the same part, and zero otherwise. He finds a heuristic solution to this approximation problem by solving a transportation problem involving the eigenvectors corresponding to the k algebraically largest eigenvalues of A.

Boppana [9] has described a spectral method using the eigenvalues and eigenvectors of the adjacency matrix which he proves will find a minimum size bisection for almost all graphs in a certain class of random graphs.

3. Lower Bounds. We obtain lower bounds on the sizes of vertex separators in terms of the eigenvalues of the Laplacian matrix $Q(G)$ in this section. The lower bounds hold for *any* vertex separator in the graph; in particular, these bounds apply to the smallest separator in the graph. We assume the graph G to be connected.

Let $G = (V, E)$ denote a graph on $|V| = n$ vertices, and let A be a subset of its vertices. Denote by $\rho(v, A)$ the distance of a vertex v from A, i.e., the fewest number of edges in a shortest path from v to a vertex in A. Let S denote the set of vertices which are at a distance less than $\rho \geq 2$ from A, and not belonging to A. Hence

$$S = \{v \in V \setminus A : \rho(v, A) < \rho\}.$$

Define $B = V \setminus (A \cup S)$; if $B \neq \emptyset$, then the distance between A and B, $\rho(A, B) = \rho$. If $\rho > 2$, the set S is a *wide separator* that separates A from B. If $\rho = 2$, we get the commonly used notion of separators. Wide separators have been used in sparse matrix algorithms by George and Ng [21], and Gilbert and Zmijewski [22].

Let E_A denote the set of edges with both endpoints in A, and E_{AS} denote the set of edges with one endpoint in A, and the other in S. The sets E_B, E_S, and E_{BS} are defined similarly. In the following, it will be convenient to work with the fractional sizes $a \equiv |A|/n$, $b \equiv |B|/n$, and $s \equiv |S|/n$. The degree of a vertex v will be denoted by $d(v)$, and Δ will denote the maximum degree of vertices in G.

The first result is a lower bound on the size of a wide separator separating any pair of vertex disjoint sets A and B which are at a distance ρ from each other.

THEOREM 2. *Let A, B be disjoint subsets of vertices of G which are at a distance $\rho \geq 2$ from each other. Let S denote the set of vertices not belonging to A which are at a distance less than ρ from A. Then*

$$s^2 + \beta s - \rho^2 a(1 - a) \geq 0, \quad \textit{where } \beta = (\Delta/\lambda_2) + \rho^2 a - 1.$$

Proof: Let $\underline{e}$, $\underline{0}$ be the vector of all ones and all zeros, respectively. The Courant-Fischer

minimax principle states that

$$\begin{aligned} \lambda_2 &= \min_{\substack{\underline{x}\neq\underline{0} \\ \underline{e}^t\underline{x}=0}} \frac{\underline{x}^t Q \underline{x}}{\underline{x}^t \underline{x}} \\ &= \min_{\substack{\underline{x}\neq\underline{0} \\ \underline{e}^t\underline{x}=0}} \frac{\sum_{(i,j)\in E}(x_i - x_j)^2}{\sum_{i=1}^n x_i^2}. \end{aligned} \tag{1}$$

Using the Lagrange identity

$$n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2 = \sum_{\substack{i,j=1 \\ i<j}}^{n} (x_i - x_j)^2,$$

Fiedler [18] derived the following inequality, valid for all non-constant n-vectors $\underline{x}$, from (1).

$$n \sum_{(i,j)\in E} (x_i - x_j)^2 \geq \lambda_2 \sum_{\substack{i,j\in V \\ i<j}} (x_i - x_j)^2. \tag{2}$$

We prove the result by making an appropriate choice of $\underline{x}$ in the above inequality.

Choose the v-th component of $\underline{x}$ be

$$x_v = 1 - \frac{2}{\rho} \min\{\rho, \rho(v, A)\}.$$

If $v \in A$, then $x_v = 1$; if $v \in B$, then $x_v = -1$; and if $v \in S$, then $-1+(2/\rho) \leq x_v \leq 1-(2/\rho)$. Also, if v, w are adjacent vertices, then $|x_v - x_w| \leq 2/\rho$.

The left-hand side of equation (2) has nonzero contributions from three terms, and it can be bounded from above as follows.

$$\begin{aligned} \sum_{(i,j)\in E} (x_i - x_j)^2 &= \left(\sum_{\substack{(i,j)\in E \\ i\in A, j\in S}} + \sum_{\substack{(i,j)\in E \\ i\in B, j\in S}} + \sum_{\substack{(i,j)\in E \\ i\in S, j\in S}} \right) (x_i - x_j)^2 \\ &\leq \frac{4}{\rho^2}(|E_{AS}| + |E_{BS}| + |E_S|) \\ &\leq \frac{4}{\rho^2} ns\Delta. \end{aligned} \tag{3}$$

Similarly, nonzero contributions to the right-hand side of equation (2) also come from three terms, and we obtain a lower bound as shown.

$$\begin{aligned} \sum_{\substack{i,j\in V \\ i<j}} (x_i - x_j)^2 &= \left(\sum_{i\in A, j\in S} + \sum_{i\in A, j\in B} + \sum_{i\in B, j\in S} \right) (x_i - x_j)^2 \\ &\geq \left(1 - (1 - \frac{2}{\rho})\right)^2 n^2 as + (1 - (-1))^2 n^2 ab + \left(-1 - (-1 + \frac{2}{\rho})\right)^2 n^2 bs \end{aligned}$$

$$= \frac{4n^2}{\rho^2}\left((a+b)s + \rho^2 a(1-a-s)\right)$$

$$= \frac{4n^2}{\rho^2}\left((1-s)s + \rho^2 a(1-a-s)\right). \tag{4}$$

Using inequalities (3) and (4) in Fiedler's inequality (2), and canceling common terms, we obtain

$$s\Delta \geq \lambda_2\left((1-s)s + \rho^2 a(1-a-s)\right).$$

After some rearrangement, this yields the desired result. ■

Fiedler [17] proved that λ_2 satisfies

$$\lambda_2 \leq (n/(n-1)) \min\{d(v) : v \in V\}.$$

Mohar [33] proved that for all graphs except the complete graph K_n, $\Delta \geq \lambda_2$. Thus for all graphs except the complete graph, the ratio $\Delta/\lambda_2 \geq 1$, and β is a positive number. Indeed, for all the adjacency graphs of sparse matrices that we have computed partitions, the ratio Δ/λ_2, and hence β, is much larger than one.

COROLLARY 3. *If $\beta \geq \rho$, then*

$$s \geq \frac{\rho^2 a(1-a)}{\beta} = \frac{\rho^2 a(1-a)}{(\Delta/\lambda_2) + \rho^2 a - 1}.$$

Proof: Let s_1, s_2 be the roots of the quadratic equation corresponding to the inequality in Theorem 2, with $s_1 < s_2$. Then $s \geq s_2$, and

$$s_2 = \frac{1}{2}\left(-\beta + (\beta^2 + 4\rho^2 a(1-a))^{1/2}\right).$$

If $\beta \geq 2\rho(a(1-a))^{1/2}$, then expanding the rhs in power series yields the result.

It remains to verify the condition of the corollary. Since $(a(1-a))^{1/2}$ has its maximum value $1/2$ when $0 \leq a \leq 1$, the power series expansion is valid when $\beta \geq \rho$. ■

The corollary exhibits the dependence of vertex separator sizes on λ_2: the smaller the second eigenvalue, the larger the ratio Δ/λ_2, and the smaller the lower bound on the vertex separator size. The corollary also shows the dependence on the lower bound on the distance ρ and the fractional size of the set A.

The common situation of a separator corresponds to $\rho = 2$. In this case, the quadratic inequality becomes

$$s^2 + \beta s - 4a(1-a) \geq 0, \quad \text{with } \beta = (\Delta/\lambda_2) + 4a - 1.$$

After some simplification, it can be seen that the inequality in Theorem 2.1 of Alon, Galil, and Milman [1] is equivalent to the above inequality. In this case, when $\beta \geq 2$, we obtain the lower bound

$$s \geq \frac{4a(1-a)}{(\Delta/\lambda_2) + 4a - 1}.$$

Mohar (Lemma 2.4, [32]) has obtained a lower bound on vertex separators in terms of the ratio λ_n/λ_2.

We can also obtain lower bounds on edge separators. By omitting the last step in equation (3), and replacing the sum $|E_{AS}| + |E_{BS}| + |E_S|$ by $|E| - |E_A| - |E_B|$, we have

$$|E| - |E_A| - |E_B| \geq \lambda_2 n \left((a+b) - (a+b)^2 + \rho^2 ab\right),$$

for two vertex disjoint sets A, B which are a distance ρ apart.

Alon and Milman [2] have obtained the inequality

$$|E| - |E_A| - |E_B| \geq \frac{\rho^2 \lambda_2 n}{((1/a) + (1/b))} = \frac{\rho^2 \lambda_2 nab}{(a+b)}.$$

These techniques can also be used to bound the number of edges separating two disjoint sets of vertices from each other.

A second lower bound. We now obtain a lower bound that exhibits another factor influencing the size of vertex separators. The technique used is derived from the Wielandt-Hoffman theorem, and has been previously used by Donath and Hoffman [12] to obtain lower bounds on edge separators.

Let S be a vertex separator that separates the graph G into two sets A and B, with $|A| \geq |B| \geq |S|$. Let $d(v)$ denote the degree of a vertex v, and let $i(v)$ denote the 'internal' degree of v, i.e., the number of edges incident on v with the other endpoint in the same set as v.

Recall that the eigenvalues of the Laplacian matrix Q are ordered as

$$\lambda_1 = 0 < \lambda_2 \leq \lambda_3 \ldots \leq \lambda_n.$$

Let the $n \times n$ matrix $J = \mathrm{diag}(J_a, J_b, J_c)$, where J_a is the $na \times na$ matrix of all ones, and J_b, J_c are similarly defined. The eigenvalues of J are

$$\mu_1 = na \geq \mu_2 = nb \geq \mu_3 = ns > \mu_4 = \ldots \mu_n = 0.$$

THEOREM 4. *Let S be a vertex separator that divides a graph G into two parts A, B, with $|A| \geq |B| \geq |S|$. Then*

$$s \geq \frac{(1-a)\lambda_2}{2\Delta - (\lambda_3 - \lambda_2)}.$$

Proof: From the proof of the Wielandt-Hoffman theorem [26] (see also [12]),

$$\mathrm{trace}(-QJ) \leq \sum_{i=1}^{n} (-\lambda_i)\mu_i.$$

Canceling out the minus signs yields

$$\mathrm{trace}(QJ) \geq \sum_{i=1}^{n} \lambda_i \mu_i. \tag{5}$$

We now compute both sides of the above inequality.

The right-hand side is

$$\begin{aligned} \sum_{i=1}^{n} \lambda_i \mu_i &= na \cdot 0 + nb \cdot \lambda_2 + ns \cdot \lambda_3 \\ &= nb\lambda_2 + ns\lambda_3 \\ &= n(1-a-s)\lambda_2 + ns\lambda_3. \end{aligned} \tag{6}$$

To evaluate the left-hand side, we partition the symmetric matrix Q to conform to J:

$$Q = \begin{pmatrix} Q_{aa} & 0 & Q_{as} \\ 0 & Q_{bb} & Q_{bs} \\ Q_{as}^t & Q_{bs}^t & Q_{ss} \end{pmatrix}.$$

$$\begin{aligned} \text{trace}(QJ) &= \text{trace}(Q_{aa}J_a) + \text{trace}(Q_{bb}J_b) + \text{trace}(Q_{ss}J_s) \\ &= \left(\sum_{v \in A} + \sum_{v \in B} + \sum_{v \in S} \right) d(v) - i(v) \\ &= 2(|E| - |E_A| - |E_B| - |E_S|) \\ &\leq 2(|E| - |E_A| - |E_B|) \\ &= 2ns\Delta. \end{aligned} \tag{7}$$

Substituting the inequalities (6) and (7) in (5), we obtain

$$2ns\Delta \geq n(1-a-s)\lambda_2 + ns\lambda_3.$$

After some rearrangement this yields the final result. ■

This last lower bound on a vertex separator size shows as before that the magnitude of λ_2 influences the lower bound; it also shows that the 'gap' between λ_3 and λ_2 has an effect. The bounds are large when the gap is large.

Donath and Hoffman [12] obtained lower bounds on the number of edges cut by the partition when the set of vertices is partitioned into k sets, in terms of the eigenvalues of a matrix $M = A + U$, where A is the adjacency matrix and U is a diagonal matrix constrained only by the requirement that its diagonal elements sum to $-2|E|$.

Finally, a word of caution is in order about these lower bounds. These bounds should be considered the same way one treats an upper bound on the error in an a priori roundoff error analysis [44] (cited in [23]). The lower bounds obtained are not likely to be tight, except for certain classes of graphs. They do illustrate, however, that a large λ_2, with an accompanying small Δ/λ_2, will result in large sizes for the best separators in a graph.

4. Partitions of grid graphs. In this section we show that the second eigenvector of the Laplacian matrix can be used to find good edge and vertex separators in grid graphs. The vertex separators obtained will be identical to the separators at the first level in a theoretical nested dissection (ND) scheme.

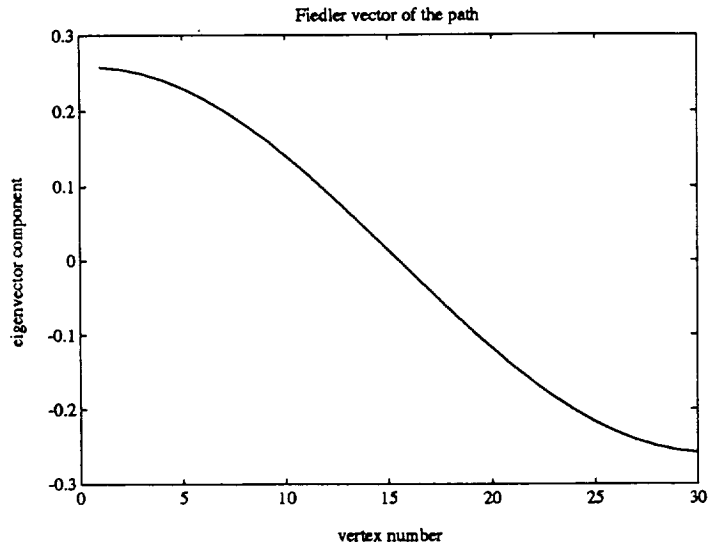

FIG. 1. *The second Laplacian eigenvector of the path.*

To compute separators by this technique, we need to obtain the eigenvectors of grids. We will show that the Laplacian spectra of grid graphs can be explicitly computed in terms of the Laplacian spectra of path graphs. Thus we can compute the second eigenvectors of the grid graphs in terms of the Laplacian eigenvectors of the path graph.

The path graph. Let P_n denote the path graph on n vertices. For concreteness, we assume in the discussion to follow that $n \geq 2$ is even. We number the vertices of the path from 1 to n in the natural order from left to right. Let $\phi_n \equiv \pi/n$. We denote the elements of a vector $\underline{x}$ by writing its i-th component as (x_i).

LEMMA 5. *The Laplacian spectrum of P_n is*

$$\begin{aligned} \lambda_{k,n} &= 4\sin^2(\frac{1}{2}(k-1)\phi_n), \\ \underline{x}_{k,n} &= (\cos((i-1/2)(k-1)\phi_n)), \quad \textit{for } k = 1,\ldots,n,\ i = 1,\ldots,n. \end{aligned}$$

As k ranges from 1 to n, the angle $(1/2)(k-1)\phi_n$ varies from zero to $\pi/2$; hence the eigenvalues are ordered as $\lambda_{1,n} \leq \lambda_{2,n} \ldots \leq \lambda_{n,n}$. Note that $\lambda_{1,n} = 0$, $\underline{x}_{1,n} = \underline{1}$, and $\lambda_{2,n} = 4\sin^2(\frac{\phi_n}{2})$, and $\underline{x}_{2,n} = (\cos((i-1/2)\phi_n))$. The components of $\underline{x}_{2,n}$ corresponding to vertices of P_{30} are shown in Fig. 1.

Let x_l denote the median ($n/2$-th largest) component of the second eigenvector, and partition the vertices of the path into two sets, one set consisting of all vertices with components less than or equal to the median component, and the other consisting of all vertices with components larger than the median component. This partitions the path into equal size subsets of vertices, one consisting of the vertices with positive eigenvector components and the other consisting of vertices with negative components.

If x_v denotes the component of the second eigenvector corresponding to vertex v, $|x_v - x_w|$ is a measure of the distance between the vertices v and w in the path: Fiedler calls this the

algebraic distance between v and w.

Graph and Kronecker products. We can compute the spectra of grid graphs from the spectra of the path graph. We require the concepts of graph products and the Kronecker products of matrices. Our notation for graph products is from Cvetkovic, Doob, and Sachs [11], and a good discussion of Kronecker products may be found in Fiedler [19] .

For $i = 1, 2$, let $G_i = (V_i, E_i)$ be graphs. The *Cartesian sum* $G_1 + G_2$ is the graph $(V_1 \times V_2, E)$, where vertices (i_1, j_1) and (i_2, j_2) are joined by an edge if either $i_1 = i_2$ and $\{j_1, j_2\}$ is an edge in G_2, or $j_1 = j_2$ and $\{i_1, i_2\}$ is an edge in G_1.

The *Cartesian product* $G_1 \cdot G_2$ is the graph $(V_1 \times V_2, F)$ where vertices (i_1, j_1) and (i_2, j_2) are joined by an edge if $\{i_1, i_2\}$ is an edge in G_1 and $\{j_1, j_2\}$ is an edge in G_2.

The *strong sum* $G_1 \oplus G_2$ is the graph $(V_1 \times V_2, E \cup F)$; thus it contains the edges in both the Cartesian sum and the Cartesian product.

It is easy to verify that the Cartesian sum $P_n + P_m$ is the five-point $m \times n$ grid graph, and that the strong sum $P_n \oplus P_m$ is the nine-point $m \times n$ grid graph.

Since the grid graphs can be obtained from appropriate graph products of the path graph, the Laplacian matrices of the $m \times n$ grid graphs can be obtained from Kronecker products involving the Laplacian matrices of the path graph. If C is a $p \times q$ matrix, and D is $r \times s$, the *Kronecker product* $C \otimes D$ is the $pr \times qs$ matrix with each element d_{ij} of D replaced by the submatrix (Cd_{ij}). (Equivalently we can consider each element c_{ij} of C replaced by the submatrix ($c_{ij}D$).)

The five-point grid. We consider the $m \times n$ five-point grid, and without loss of generality consider $m \leq n$. For concreteness, initially we consider the case when n is even, and $m < n$. At the end of this section, we discuss the case when n is odd, or $m = n$. We draw the $m \times n$ grid with n vertices in each row and m vertices in each column.

Let Q denote the Laplacian matrix of the five-point $m \times n$ grid graph, R_n denote the Laplacian matrix of the path graph on n vertices, and I_n be the identity matrix of order n. Recall that $\lambda_{k,n}$, $\underline{x}_{k,n}$ denotes the k-th eigenpair (when eigenvalues are listed in increasing order) of the path graph with n vertices. The following result is well-known; we include a proof for completeness, and because we wish to indicate how a similar result is obtained for the Laplacian spectrum of a modified nine-point grid.

THEOREM 6. *The Laplacian spectrum of the $m \times n$ five-point grid is*

$$\begin{aligned} \mu_{k,l} &= \lambda_{k,n} + \lambda_{l,m}, \\ \underline{y}_{k,l} &= \underline{x}_{k,n} \otimes \underline{x}_{l,m}, \quad k = 1, \ldots, n, \ l = 1, \ldots, m. \end{aligned}$$

Proof: It is easy to verify that the Laplacian matrix of the five-point grid can be expressed in terms of the Laplacian matrix of the path graph as

$$Q = R_n \otimes I_m \ + \ I_n \otimes R_m.$$

The first term in the sum creates m copies of the path on n vertices, and the second term adds the 'vertical' edges which join neighboring vertices in each column of the grid.

We show that $\lambda_{k,l}$, $\underline{x}_{k,l}$ is an eigenpair of Q.

$$\begin{aligned} Q\underline{x}_{k,n} \otimes \underline{x}_{l,m} &= (R_n \otimes I_m)\,(\underline{x}_{k,n} \otimes \underline{x}_{l,m}) \;+\; (I_n \otimes R_m)\,(\underline{x}_{k,n} \otimes \underline{x}_{l,m}) \\ &= (R_n\underline{x}_{k,n}) \;\otimes\; (I_m\underline{x}_{l,m}) \;+\; (I_n\underline{x}_{k,n}) \;\otimes\; (R_m\underline{x}_{l,m}) \\ &= \lambda_{k,n}\underline{x}_{k,n} \otimes \underline{x}_{l,m} \;+\; \underline{x}_{k,n} \otimes \lambda_{l,m}\underline{x}_{l,m} \\ &= (\lambda_{k,n} + \lambda_{l,m})\underline{x}_{k,n} \otimes \underline{x}_{l,m}. \end{aligned}$$

The transformation from the first line to the second line uses the associativity of the Kronecker product. ■

The smallest eigenvalue $\mu_{1,1} = \lambda_{1,n} + \lambda_{1,m}$ is zero, since Q is positive semidefinite. The next smallest eigenvalue is either $\mu_{2,1}$ or $\mu_{1,2}$. Since

$$\begin{aligned} \mu_{2,1} &= \lambda_{2,n} + \lambda_{1,m} = 4\ \sin^2(\frac{\phi_n}{2}), \quad \text{and} \\ \mu_{1,2} &= \lambda_{1,n} + \lambda_{2,m} = 4\ \sin^2(\frac{\phi_m}{2}), \end{aligned}$$

and $m \leq n$, the second smallest eigenvalue is $\mu_{2,1}$. The corresponding eigenvector is

$$\underline{y}_{2,1} = \underline{x}_{2,n} \otimes \underline{x}_{1,m} = (\ \cos(i - 1/2)\phi_n\ ) \otimes \underline{1}.$$

The components of the $\underline{y}_{2,1}$ are constant along each column of m vertices, and the components decrease from left to right across a row. Columns numbered 1 to $n/2$ have positive components, and the rest of the columns have negative components. The second eigenvector components for the $m \times n$ five-point grid are shown in Figure 2.

These results show that the second eigenvector of the grid can be used to compute good edge separators and vertex separators. Let $\underline{y}$ denote this eigenvector in the following discussion, and let y_l denote the median component ($(mn/2)$-th largest component out of mn). From the preceding discussion, y_l corresponds to the components of the vertices on the $n/2$-th column. Let y_v denote the eigenvector component corresponding to vertex v.

COROLLARY 7. *Let V denote the set of vertices of the five-point $m \times n$ grid ($m < n$, n even), and let V be partitioned by its second eigenvector as follows:*

$$A' = \{v : y_v \leq y_l\}, \quad B' = V \setminus A.$$

If E' denotes the set of edges joining A' to B', then E' is an edge separator of size m which separates the grid into two parts each with $(mn/2)$ vertices. Further, if S denotes the set of endpoints of E' which belong to B', then S is a vertex separator of size m which separates the grid into two parts of $(mn/2)$ and $m((n/2) - 1)$ vertices.

The Corollary follows from noting that A' consists of vertices in the columns 1 to $n/2$ of the grid, and B' is the remaining set of columns. The edge separator E' consists of the m edges of the grid which join vertices in column $n/2$ to column $(n/2) + 1$. Finally, the vertex separator S consists of vertices in column $(n/2) + 1$. Note that the vertex separator is the same as the separator at the first level found by theoretical nested dissection. Buser [10] has shown that the edge separator E' yields the optimal isoperimetric number for grid graphs.

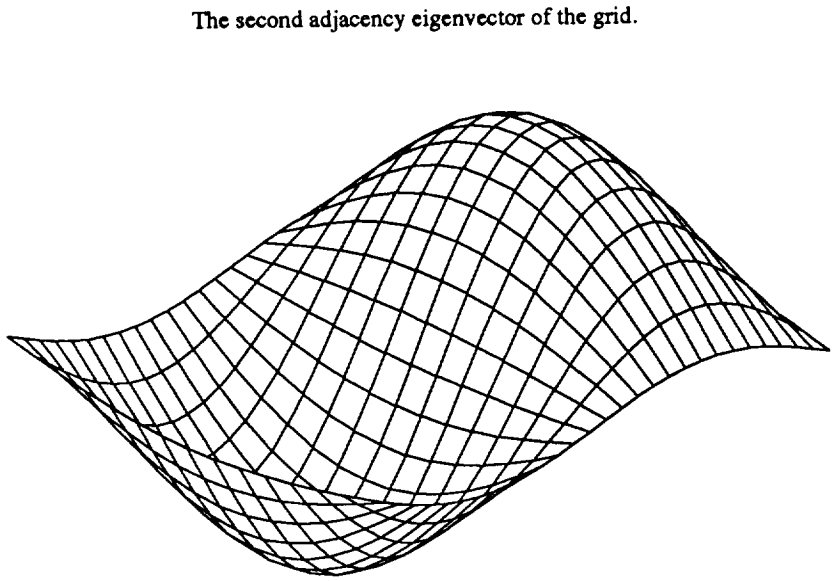

FIG. 2. *The second Laplacian eigenvector of the five-point grid.*

We now consider the two special cases. One corresponds to the number of columns n being odd. Then vertices in the middle column ($(n+1)/2$-th column) have eigenvector components equal to zero; columns numbered smaller than the middle column have positive components, and columns numbered higher have negative components. Again, across each row, the components decrease from left to right.

The second case corresponds to a square grid, $m = n$. Then $\mu_{2,1} = \mu_{1,2}$, and the second smallest eigenvalue of Q has geometric multiplicity two. The two linearly independent eigenvectors obtained by the graph product approach are

$$\begin{aligned} \underline{y}_{2,1} &= \underline{x}_{2,n} \otimes \underline{x}_{1,n} \\ \underline{y}_{1,2} &= \underline{x}_{1,n} \otimes \underline{x}_{2,n}. \end{aligned}$$

The eigenvector $\underline{y}_{2,1}$ has components as described earlier for the rectangular case, with components constant along each column, and the components decreasing from positive to negative values across each row. If n is odd, the middle column has its components equal to zero. The eigenvector $\underline{y}_{1,2}$ has components constant across each row, and decreasing from bottom to top along each column. If n is odd, the middle row has components zero, and rows below it have positive components, and rows above have negative components. If n is even, rows one to $n/2$ have negative components, and the rest have positive components. From these two independent eigenvectors, we obtain a middle column and a middle row as the vertex separators.

The nine-point grid. Let Q' denote the Laplacian matrix of the nine-point grid, and let

D_n be the $n \times n$ diagonal degree matrix of the n-vertex path. As before, let R_n denote the Laplacian matrix of the n-vertex path, and I_n the identity matrix of order n. It is again not difficult to verify that

$$Q' = R_n \otimes I_m + I_n \otimes R_m + R_n \otimes D_m + D_n \otimes R_m - R_n \otimes R_m.$$

Unfortunately, the spectrum of Q' can not be expressed in terms of the spectra of the path graphs, as for the five-point grid.

However, we can first embed the nine-point grid graph in a modified grid, whose Laplacian spectrum is computable in terms of the spectra of the path graphs, and then partition the modified grid. We use the partition of the modified grid to partition the nine-point grid.

The necessary modification to the nine-point grid is as follows. Replace each boundary edge of the $m \times n$ grid by *two* edges joining the same endpoints. Let Q denote the Laplacian of the resulting multigraph.

THEOREM 8. *The spectrum of Q is*

$$\begin{aligned} \mu_{k,l} &= 3(\lambda_{k,n} + \lambda_{l,m}) - \lambda_{k,n}\,\lambda_{l,m} \\ \underline{y}_{k,l} &= \underline{x}_{k,n} \otimes \underline{x}_{l,m}, \quad \textit{for } k = 1,\ldots,n,\ l = 1,\ldots,m. \end{aligned}$$

Proof: It is easy to show that

$$Q = 3(R_n \otimes I_m + I_n \otimes R_m) - R_n \otimes R_m.$$

A direct computation, as in Theorem 6, shows that $\mu_{k,l}$, $\underline{y}_{k,l}$ is an eigenpair of Q. ■

Note that the eigenvectors of the modified nine-point grid are the same as the eigenvectors of the five-point grid. As in the five-point grid, the smallest eigenvalue $\mu_{1,1}$ is zero; the second smallest eigenvalue is $\mu_{2,1} = 12\ \sin^2(\phi_n/2)$. The eigenvector $\underline{y}_{2,1}$ is the second eigenvector of the five-point grid. Hence the partitions of the modified nine-point grid are exactly the same as those of the five-point grid.

Adjacency spectra of paths and grids. The adjacency spectra of grid graphs can also be computed in terms of the adjacency spectrum of the path. Let A_n denote the adjacency matrix of the path on n vertices, P_n, and denote $\theta_n \equiv \pi/(n+1)$. The adjacency spectrum of P_n is

$$\begin{aligned} \lambda_{k,n} &= 2\ \cos(k\,\theta_n), \\ \underline{x}_{k,n} &= (\ \sin(k\,i\,\theta_n)\ ), \quad \text{for } k = 1,\ldots,n,\ l = 1,\ldots,m. \end{aligned}$$

Since θ_n ranges from zero to π as k varies from 1 to n, the eigenvalues are ordered

$$\lambda_{1,n} \geq \lambda_{2,n} \ldots \geq \lambda_{n,n}.$$

The principal eigenvector $\underline{x}_{1,n} = (\ \sin(i\theta_n)\ )$ has all positive components, as required by Perron-Frobenius theory. A plot of the components of this vector against vertex numbers shows a half sine curve, with its maximum near $n/2$, and near zero components at either ends.

The eigenvector corresponding to the second largest eigenvector is $\underline{x}_{2,n} = (\ \sin(2i\theta_n)\ )$, and a plot of this vector against vertex numbers shows a full sine curve. It attains its maximum near $n/4$, its minimum near $3n/4$, and has near zero values at either end, and near $n/2$.

Recall that the components of the Fiedler vector of the path decrease monotonically from left to right along the path. Hence a partition of the vertices with respect to some component of the Fiedler vector yields two parts, each forming a connected subgraph. This may not be true of a partition with respect to the second eigenvector of the adjacency spectrum, because of its sinusoidal behavior.

Barnes's algorithm [3] for partitioning vertices into two sets such that a small edge separator joins the sets makes use of the first two adjacency eigenvectors. If we apply Barnes's algorithm to partition the path P_{30} into two sets of half the vertices each, one part consists of the first thirteen vertices, and the last two vertices. This part has two connected components. The second part consisting of vertices 14 to 28 forms a connected subgraph.

Now we consider the adjacency spectra of the grids. Let B, C denote the adjacency matrices of the five and nine-point $m \times n$ grids, respectively. It can be verified that

$$\begin{aligned} B &= A_n \otimes I_m + I_n \otimes A_m, \\ C &= A_n \otimes I_m + I_n \otimes A_m + A_n \otimes A_m. \end{aligned}$$

The eigenvalues of B are

$$\mu_{k,l} = \lambda_{k,n} + \lambda_{l,m}, \quad \text{for } k = 1, \ldots, n,\ l = 1, \ldots, m,$$

while those of C are

$$\mu_{k,l} = \lambda_{k,n} + \lambda_{l,m} + \lambda_{k,n}\lambda_{l,m}, \quad \text{for } k = 1, \ldots, n,\ l = 1, \ldots, m.$$

The corresponding eigenvectors for both grids are

$$\underline{y}_{k,l} = \underline{x}_{k,n} \otimes \underline{x}_{l,m}.$$

Again, for both grids, the largest eigenvalue is $\mu_{1,1}$, and the the second largest is $\mu_{2,1}$. The corresponding eigenvector $\underline{y}_{2,1}$ is plotted in Figure 3.

Since a plot of the components of the second eigenvector across a row yields a full sinusoidal curve, as in the case of the path, a partition of the vertices into two sets with respect to some component of this vector may result in parts consisting of disconnected subgraphs. Barnes's algorithm applied to the 15×30 grid yields two connected components in one part, and a single component in the other part.

5. A spectral partitioning algorithm. In this section we describe an algorithm for finding a vertex separator of a graph by means of its Lapacian matrix. Recall that we require the separator to partition the graph into two parts with nearly equal numbers of vertices in each part, and also that the size of the vertex separator be small.

The algorithm uses the second eigenvector of the Laplacian matrix to compute the partition. We find x_m, the median value of the components of the eigenvector. Let A' be the set of vertices whose components are less than or equal to x_m, and let B' be the remaining

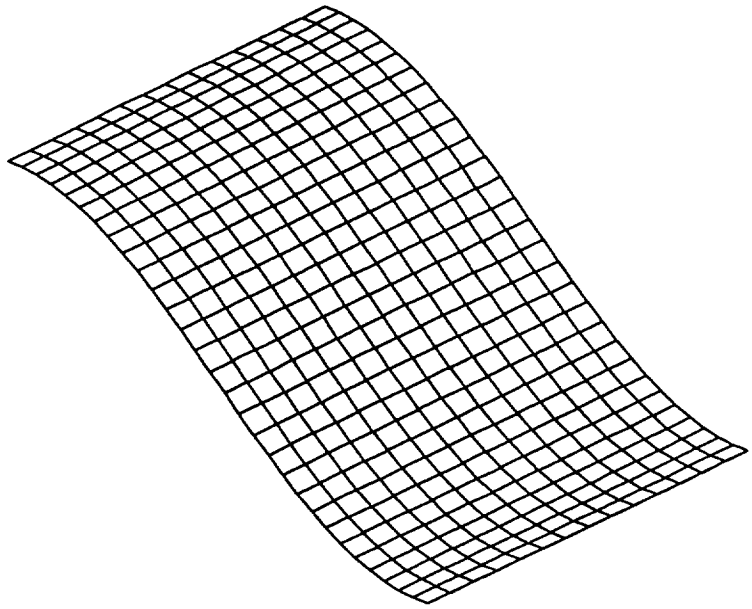

FIG. 3. *The second adjacency eigenvector of the five- and nine-point grids.*

set of vertices. If there is a single vertex with the component corresponding to x_m, then A' and B' differ in size by at most one. If there are several vertices with components equal to x_m, arbitrarily assign such vertices to A' or B' to make these sets differ in size by at most one.

This initial partition of G gives an edge separator in the graph. Let A_1 denote the vertices in A' which are adjacent to some vertex in B', and similarly let B_1 be the set of vertices in B' which are adjacent to some vertex in A'. Let E_1 be the set of edges of G with one endpoint in A_1 and the other in B_1. Then E_1 is an edge separator of G. Note that the subgraph $H = (A_1, B_1, E_1)$ is bipartite.

We require a vertex separator of G, which can be obtained from the edge separator E_1 by several methods. The simplest method is to choose the smaller of the two endpoint sets A_1 and B_1. Gilbert and Zmijewski [22] have computed vertex separators from edge separators in this manner in the context of a parallel Kernighan-Lin algorithm. Leiserson and Lewis [29] have computed vertex separators from edge separators by a heuristic algorithm. However, there is a better way to choose a *smallest* vertex separator which can be computed from the given edge separator E'.

The idea is to choose a set S consisting of some vertices from *both* sets of endpoints A_1 and B_1, such that every edge in E_1 is incident on at least one of the vertices in S. The set S is a vertex separator in the graph G since the removal of these vertices causes the deletion of all edges incident on them, and this latter set of edges contains the edge separator E_1. The set S is a *vertex cover* (*cover*) of the bipartite graph H.

A cover of smallest cardinality is a *minimum cover*. A minimum cover S of the graph H is a smallest vertex separator of G corresponding to the edge separator E_1. The separator S can be computed by finding a maximum matching in H. Details are provided later in this section.

In general, S will consist of vertices from both A_1 and B_1. Let A_s and B_s denote the vertices of S that belong to A_1 and B_1, respectively. Then S separates G into two subgraphs with vertex sets $A = A' \setminus A_s$, $B = B' \setminus B_s$. The *Spectral Partitioning Algorithm* is summarized in Figure 4.

1. Compute the eigenvector $\underline{x}_2$ and the median value x_m of its components;
2. Partition the vertices of G into two sets:
   $A' = \{\text{vertices with } x_v \leq x_m\}$;
   $B' = V \setminus A$;
   If $|A'| - |B'| > 1$, move enough vertices with components equal to x_m
   from A' to B' to make this difference at most one;
3. Let A_1 be the set of vertices in A' adjacent to some vertex in B';
   Let B_1 be the set of vertices in B' adjacent to some vertex in A';
   Compute $H = (A_1, B_1, E_1)$, the bipartite subgraph induced by the vertex sets A_1, B_1;
4. Find a minimum vertex cover S of H by a maximum matching;
   Let $S = A_s \cup B_s$, where $A_s \subseteq A_1$, $B_s \subseteq B_1$;
   S is the desired vertex separator, and separates G into subgraphs with
   vertex sets $A = A' \setminus A_s$, $B = B' \setminus B_s$.

FIG. 4. *The spectral partitioning algorithm*

Computing a minimum cover. We now describe how a minimum cover S of the bipartite graph H is computed by matching techniques. In the following description, call one set of vertices of the bipartite graph 'rows' and the other set 'columns'. A *matching* in a graph is a subset of its edges which do not share a common endpoint. A *maximum matching* is a matching of maximum cardinality. A vertex is *matched* if it is an endpoint of an edge in a matching, and *unmatched* otherwise. A bipartite graph has a *perfect matching* if all rows and columns are matched in a maximum matching; necessarily, the number of rows must be equal to be the number of columns. Figure 5 shows a maximum matching in a bipartite graph. The matched edges are indicated with thick lines. There are several efficient algorithms to compute maximum matchings in graphs.

A *path* in a graph is a sequence of distinct vertices $v_1, v_2, \ldots, v_k$ such that (v_i, v_{i+1}) is an edge for $i = 1, \ldots, k-1$. An *alternating path* with respect to a matching M is a path which has alternate edges in M; by definition of a matching, then the remaining edges do not belong to M.

With respect to a maximum matching M in a bipartite graph H, we can define the following sets. The reader may find Figure 5 helpful to understand these definitions. Let

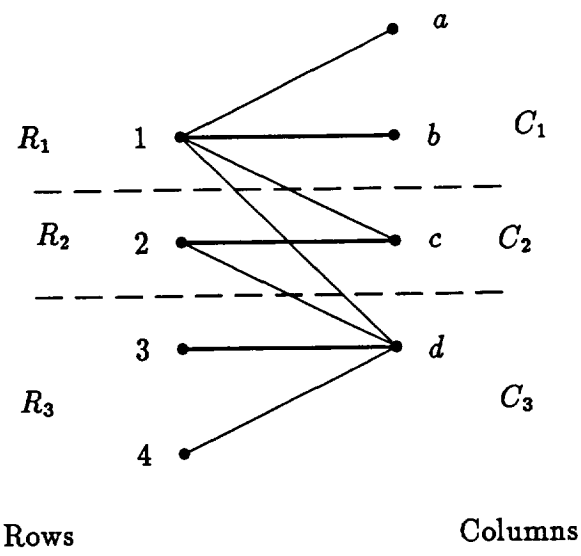

FIG. 5. *A maximum matching and a minimum cover in a bipartite graph.*

R_1 (C_1) denote the set of rows (columns) reachable by alternating path from all unmatched columns. All rows in R_1 are matched since M is a maximum matching. We include all unmatched columns in the set C_1, since these columns are reachable by alternating paths of length zero. Similarly, let C_3 (R_3) denote the set of columns (rows) reachable by alternating path from all unmatched rows. Again, all columns in C_3 are matched, and R_3 includes all unmatched rows. There may be rows and columns that have not been visited by alternating paths; include such columns in C_2 and rows in R_2. ¿From the definitions of C_1 and R_3, it can be shown that all vertices in R_2 and C_2 are matched, these sets have the same cardinality, and that they are perfectly matched to each other.

It should be noted that in a given bipartite graph, some of these sets may be absent. For instance, if there are no unmatched columns in a maximum matching, then the sets C_1 and R_1 will be absent. Similarly, if there are no unmatched rows, R_3 and C_3 will be absent. If alternating paths from unmatched rows and unmatched columns visit all rows and columns, the sets R_2 and C_2 will be empty.

Maximum matchings and minimum covers are dual concepts in bipartite graphs, and a minimum cover S of H can be obtained as either $R_1 \cup R_2 \cup C_3$, or $R_1 \cup C_2 \cup C_3$. These provide two different vertex separators in the original graph G, if R_2 and C_2 are not empty, and this freedom can be used to choose a separator that makes the parts A and B less unequal. In the bipartite graph in Fig. 5, a minimum cover is either the set $\{1, 2, d\}$ or the set $\{1, c, d\}$ with cardinality three.

These row sets and column sets can be used to compute a canonical decomposition of bipartite graphs called the Dulmage-Mendelsohn decomposition. This decomposition induces a block triangular form of sparse matrices, which is useful in several sparse matrix algorithms. A detailed description may be found in [39].

Complexity of the algorithm. Now we consider the complexity of each step of this algorithm.

The dominant cost is the cost of the Lanczos algorithm for computing the second eigen-

TABLE 1
Partitions using median component of the second Laplacian eigenvector.

key	edge separator			vertex separators					
				endpoint set			matching		
	$\|E_1\|$	$\|A'\|$	$\|B'\|$	$\|A_1\|$	$\|A'\| - \|A_1\|$	$\|B'\|$	$\|S\|$	$\|A\|$	$\|B\|$
BCSPWR09	34	862	861	22	840	861	20	857	846
BCSPWR10	44	2,650	2,650	35	2,615	2,650	31	2,623	2,646
BCSSTK13	3,585	1,002	1,001	295	707	1,001	236	862	905
CAN 1072	165	536	536	53	483	536	33	525	514
DWT 2680	85	1,340	1,340	29	1,311	1,340	28	1,313	1,339
JAGMESH	50	468	468	26	442	468	26	442	468
LSHP3466	121	1,733	1,733	61	1,672	1,733	61	1,672	1,733
NASA1824	740	912	912	103	809	912	102	839	883
NASA2146	934	1,073	1,073	96	977	1,073	74	1,036	1,036
NASA4704	1,324	2,352	2,352	185	2,167	2,352	172	2,266	2,266
GRD61.101.5	61	3,111	3,050	61	3,050	3,050	61	3,050	3,050
GRD61.101.9	181	3,111	3,050	61	3,050	3,050	61	3,050	3,050
GRD80.80.5	80	3,200	3,200	80	3,120	3,200	80	3,120	3,200
GRD80.80.9	238	3,200	3,200	80	3,120	3,200	80	3,120	3,200

vector, which costs $O(e)$ flops per iteration.

We find the median component of the second eigenvector of the Laplacian matrix by an algorithm that selects the k-th element out of n. This algorithm finds the desired element by repeatedly partitioning a subarray with respect to a pivot element, and does not sort the array. This algorithm is $O(n)$ in the average case, and $\Omega(n^2)$ in the worst case. The selection of the k-th element can be done in linear time in the worst case by an algorithm designed by Blum, Floyd, Pratt, Rivest, and Tarjan [8].

The partition into the sets A and B can be done in $O(n)$ time. The bipartite graph H can be generated in $O(e)$ time, by examining the adjacency list of each vertex at most once. Let m be the smaller of $|A'|$ and $|B'|$, and let $e' \equiv |E'|$. A maximum matching and the minimum cover S can be obtained in $O(\sqrt{m}e')$ time by an algorithm of Hopcroft and Karp [27]; this algorithm has been efficiently implemented by Duff and Wiberg [14].

6. Results. In this section, we report computational results obtained from the *Spectral Partitioning Algorithm* and provide comparisons with the *Nested Dissection Algorithm*, the *Kernighan-Lin Algorithm*, the *Fiduccia-Mattheyses Algorithm*as implemented by Leiserson and Lewis [29], and the vertex partitioning based on the multiple minimum degree algorithm by Liu [31]. Several sparse matrices from the Boeing-Harwell collection [13] and five- and nine-point grids are partitioned using these algorithms.

The partitions obtained with the *Spectral Partitioning Algorithm* are tabulated in Table 1. The edge separator E_1 separates the graph into two parts A' and B'. The sizes of these

sets are shown in the first group of three columns in the table. We show two vertex separators obtained from E_1: the first vertex separator is chosen to be the the smaller endpoint set of E_1; in the table, this set is denoted A_1. The second vertex separator S includes subsets of vertices from both endpoint sets, and is computed by means of a maximum matching to be a minimum vertex cover of the bipartite graph induced by E_1.

For the Boeing-Harwell problems, the matching method computes vertex separators that are, on the average, about 11.5 percent smaller than the separators obtained from the smaller endpoint set. Since there are two choices for the minimum cover, a good choice also makes the two part sizes less different.

The edge separators obtained are small relative to the total number of edges in each graph, except for the BCSSTK13 problem, which has a high average degree. For all problems, except two, the vertex separators obtained are also relatively small ($s < 0.04$) in comparison to the parts generated by the separators. The exceptions are BCSSTK13 and NASA1824. Both these problems have large second eigenvalue λ_2. For BCSSTK13, $\lambda_2 \approx 0.65$; in contrast, for the 80×80 nine-point grid, which has good separators, $\lambda_2 \approx 4.6 \times 10^{-3}$.

For the grid graphs, vertex separators can be computed by explicitly computing the second eigenvector by the methods in Section 4. Here, we investigate the partitions obtained by the *Spectral Partitioning Algorithm* with this eigenvector computed by the Lanczos algorithm. We partitioned the 61×101 grids initially into two sets with 3050 (50 columns) and 3111 (51 columns) vertices. The edge separator obtained joins vertices in the fiftieth column to vertices in the fifty first column. The vertex separator computed is the middle (fifty first) column.

In the square grids, the second eigenvalue has geometric multiplicity two, and there are two linearly independent eigenvectors. The eigenvectors in Section 4, $\underline{y}_{2,1}$, and $\underline{y}_{1,2}$, obtained by the Kronecker products of the Laplacian eigenvectors of the path, can be used to compute two sets of edge separators. One edge separator joins vertices in the fortieth column to vertices in the forty first column, and the other joins vertices in the fortieth row to the forty first row. In general, the Lanczos algorithm will compute a linear combination of the two eigenvectors described above, leading to a different (and larger) edge separator. However, with the starting vector we used, the Lanczos algorithm converged to the eigenvector $\underline{y}_{1,2}$, and the latter edge separator was computed. The computed vertex separator consists of the vertices in the fortieth row.

We are interested in comparing the quality of the separators computed by the spectral algorithm with the separators computed from a modified Automated Nested Dissection algorithm. The nested dissection routine in SPARSPAK, FNSEP, finds a pseudo-peripheral vertex in the graph, and generates a level structure from it. It then chooses the median level in the level structure as the vertex separator. However, this choice may separate the graph into widely disparate parts. We modified this routine such that the vertex separator is chosen to be the smallest level k such that the first k levels together contain more than half the vertices. A vertex separator is obtained by removing from the vertices in level k those vertices which are not adjacent to any vertex in level $k+1$. By the construction of the level structure, the removed vertices are adjacent to vertices in level $k-1$, and hence these are added to the part containing vertices in the first $k-1$ levels. The other part has vertices

TABLE 2
Partitions from Automated Nested Dissection.

key	vertex separator			edge separators					
	$\|S\|$	$\|A\|$	$\|B\|$	$\|E_1\|$	$\|A\|$	$\|B \cup S\|$	$\|E_2\|$	$\|A \cup S\|$	$\|B\|$
BCSPWR09	68	762	893	80	762	961	130	830	893
BCSPWR10	169	2,421	2,710	209	2,421	2,879	317	2,590	2,710
BCSSTK13	302	764	937	3,035	764	1,239	4,792	1,066	937
CAN 1072	64	478	530	108	478	594	342	542	530
DWT 2680	28	1,327	1,325	84	1,327	1,353	84	1,355	1,325
JAGMESH	26	455	455	50	455	481	50	481	455
LSHP3466	59	1,711	1,696	118	1,711	1,755	116	1,770	1,696
NASA1824	137	839	848	910	839	985	1,347	976	848
NASA2146	131	1,008	1,007	1,473	1,008	1,138	1,569	1,139	1,007
NASA4704	296	2,245	2,163	2,134	2,245	2,459	2,424	2,541	2,163
GRD61.101.5	61	3,050	3,050	121	3,050	3,111	121	3,111	3,050
GRD61.101.9	111	3,025	3,025	327	3,025	3,131	333	3,131	3,025
GRD80.80.5	80	3,160	3,160	158	3,160	3,240	158	3,240	3,160
GRD80.80.9	113	3,136	3,151	333	3,136	3,264	339	3,249	3,151

in levels $k+1$ and higher. Statistics about the vertex separators so obtained are shown in Table 2.

We can also obtain two edge separators using the level structure from the set of edges joining the vertex separator to the two parts A and B. The sizes of the edge separators and the part sizes they generate are also shown in Table 2.

The *Spectral Partitioning Algorithm* computes smaller vertex separators than the *Nested Dissection Algorithm*; on the average, the spectral vertex separators are about 65 percent of the nested dissection vertex separators. The spectral algorithm also succeeds in keeping the part sizes less disparate than the latter algorithm. The average difference in the part sizes is about 7 percent for nested dissection, but there are problems for which this difference is greater than 20 percent.

For most problems, the spectral algorithm also finds smaller edge separators in the graph than nested dissection. There are a few problems where the best edge separator obtained by nested dissection is smaller than that obtained by the spectral algorithm, but the former edge separators separate the graph into parts with widely differing sizes. In the spectral algorithm, equal part sizes can be obtained by partitioning with respect to the median eigenvector component; any other choice of part sizes can also be obtained by partitioning with respect to the appropriate component. Since edge separators are computed in nested dissection by means of a level structure, part sizes cannot be controlled as effectively.

For the five-point grids, the vertex separators obtained by the *Nested Dissection Algorithm* correspond to a 'diagonal' of the grid. For the nine point grids, the algorithm fails to

find the smallest vertex separators. For all the grid problems, automated nested dissection fails to find the optimal edge separators.

6.1. Kernighan-Lin Algorithm. The *Kernighan-Lin Algorithm* is a heuristic algorithm that can be used to compute small edge separators. We investigated the use of this algorithm separately and in conjunction with the *Spectral Partitioning Algorithm*, to compute edge and vertex separators.

The Kernighan-Lin algorithm begins with an initial partition of the graph into two subsets A', B' which differ in their sizes by at most one. At each iteration, the algorithm chooses two subsets of equal size to swap between A and B, thereby reducing the number of edges which join A to B. We refer the reader to Kernighan and Lin [28], or Gilbert and Zmijewski [22] for a detailed description of how the algorithm chooses the subsets to be swapped. The algorithm terminates when it is no longer possible to decrease the size of the edge separator by swapping subsets.

One initial partition we could use is the edge partition obtained from the *Spectral Partitioning Algorithm*, and a second choice is to use a randomly computed initial partition. We consider the four graphs with the largest edge separators from Table 1, and report the sizes of the edge and vertex separators obtained with the Kernighan-Lin algorithm in Table 3. An edge separator was computed first, and then a vertex separator was obtained as before by matching methods. The column labeled 'SP' corresponds to the output of the spectral algorithm, 'SP,KL' corresponds to the Kernighan-Lin algorithm with initial partition from the spectral algorithm, and 'KL' corresponds to the Kernighan-Lin algorithm with a random initial partition.

Application of the Kernighan-Lin algorithm with the spectral partition as input succeeds in reducing the sizes of the edge separator considerably for two of the four problems. Thus if one is primarily concerned with small edge separators, applying the Kernighan-Lin algorithm to the partition produced by spectral algorithm could be worthwhile. However, the size of the vertex separator is not improved. For two of the problems, the size remains the same; for a third, it decreases by one, and the size increases for a fourth problem. Also, for two of the four problems, the spectral algorithm by itself finds better vertex separators than those obtained by the Kernighan-Lin algorithm alone.

Gilbert and Zmijewski [22] have observed that the quality of the partition found by the Kernighan-Lin algorithm strongly depends on the quality of the initial partition. They show for a grid graph that it is possible to choose a bad initial partition for the Kernighan-Lin algorithm such that the algorithm will not find a minimum edge separator.

Edge separators obtained from the Kernighan-Lin algorithm with initial spectral partition are better than those obtained from the application of the Kernighan-Lin algorithm with random initial partitions for two of the four problems. Use of the initial partition from spectral algorithm also helps the Kernighan-Lin algorithm to converge faster. On these four problems, the Kernighan-Lin algorithm ran on the average about 3.2 times faster when the spectral partition was used. Thus the spectral algorithm could be used to generate initial partitions of high quality for the Kernighan-Lin algorithm.

TABLE 3

Partitions from the Kernighan-Lin Algorithm. The first table describes the edge separators, and the second, vertex separators.

key	$\|A'\|$	$\|B'\|$	$\|E_1\|$		
			SP	SP,KL	KL
BCSSTK13	1,002	1,001	3,585	2,880	3,550
NASA1824	912	912	740	739	739
NASA2146	1,073	1,073	934	870	870
NASA4704	2,352	2,352	1,324	1,313	1,525

key	SP			SP,KL			KL		
	$\|S\|$	$\|A\|$	$\|B\|$	$\|S\|$	$\|A\|$	$\|B\|$	$\|S\|$	$\|A\|$	$\|B\|$
BCSSTK13	236	862	905	250	870	883	284	772	947
NASA1824	103	839	883	102	830	892	102	830	892
NASA2146	74	1,036	1,036	74	1,036	1,036	74	1,036	1,036
NASA4704	172	2,266	2,266	172	2,266	2,266	204	2,163	2,337

6.2. Comparison with Leiserson-Lewis' and Liu's results. In [29] Leiserson and Lewis have used the *Fiduccia-Mattheyses Algorithm* [16] to compute vertex separators and then to order sparse matrices. Liu [31] uses the multiple minimum degree ordering algorithm to compute vertex separators, and then improves the separator (by decreasing its size and making the parts less unequal) by a matching technique. He uses his separator algorithm in [30] to compute a good ordering for parallel factorization. In both implementations sparse matrices from the Boeing-Harwell Collection are used, so we are able to give a direct comparison of the top level vertex bisector here. The figures in Table 4 are obtained directly from Liu's report [31] and from Lewis (personal communication). In both cases we have added small disconnected components, which were created by the vertex separators, to the smaller of the two sets $|A|$ or $|B|$. For the Leiserson-Lewis results we list in some instances two different partitions, where it was difficult to judge which one should be considered better.

The results in Table 4 show that the current sparse matrix algorithms yield very good partitions, which generally can be judged to better than the ones produced by the *Spectral Partitioning Algorithm.* However, neither one of the two algorithms offers any easy prospect of a parallel implementation. Another factor which cannot be evaluated in this comparison is the relative excecution time of the algorithms.

7. Convergence. The dominant computation in the *Spectral Partitioning Algorithm* is the computation of the second eigenvector of the Laplacian matrix by the Lanczos algorithm. Since the Lanczos algorithm is an iterative algorithm, the number of iterations and the time

TABLE 4
Vertex separators from the Leiserson-Lewis and the Liu algorithms.

key	Leiserson-Lewis			Liu		
	$\|S\|$	$\|A\|$	$\|B\|$	$\|S\|$	$\|A\|$	$\|B\|$
BCSPWR09	7	858	854	8	1026	689
BCSPWR10	18	2641	2634	19	2661	2620
BCSSTK13	228	1001	774	298	941	764
	242	892	869			
CAN10724	31	536	506	38	665	368
	34	522	516			
DWT2680	28	1339	1313	26	1369	1283
LSHP3466	46	2381	1139	61	1727	1678
	57	1708	1701			

required to compute this eigenvector is dependent on the number of correct digits needed in the eigenvector components. In this section, we describe the details of an implementation of the Lanczos algorithm for computing this eigenvector, and study how the quality of computed separators depend on the accuracy in the second eigenvector.

7.1. The Lanczos algorithm. The most efficient algorithm for computing a few eigenvalues and eigenvectors of large, sparse symmetric matrices is the Lanczos algorithm. Since the Lanczos algorithm is discussed extensively in the textbook literature [23, 35], we do not include a detailed description of the standard algorithm here. The convergence of the Lanczos algorithm depends critically on the distribution of the eigenvalues of the underlying matrix. Usually the extreme eigenpairs, i.e. the largest and smallest, are found first. However it is also known that for operators such as the discrete Laplacian for a grid problem, or more generally for positive definite finite element matrices which are approximations to elliptic operators, the Lanczos algorithms converges in most cases to the extreme right, i.e. the very large eigenvalues, before delivering good approximations to the eigenvalues close to zero. This behavior can be explained with the so-called Kaniel-Paige-Saad theory (see [35]). When computing the smallest positive eigenvalue of the Laplacian matrix Q, one faces exactly the same situation: the Lanczos algorithm delivers very good approximations to the large eigenvalues before converging to the desired second smallest eigenvalue. Thus the Lanczos algorithm potentially requires long runs before it computes an approximation to the second eigenpair.

Several potential modifications can be incorporated in the simple Lanczos algorithm for the faster computation of the second eigenvector, but unfortunately none of them is applicable here. The first obvious improvement of the situation would be to apply the shifted and inverted operator, i.e. to consider the eigenvalue problem

$$(Q - \sigma I)^{-1} u = \mu u. \tag{8}$$

This is a standard technique in finite element applications [24], and has been used very successfully in a variety of implementations of the Lanczos algorithm [15, 25, 36, 42]. The

eigenvectors remain the same, and the eigenvalues μ of (8) are related to the eigenvalues λ of Q by a simple transformation. In the situation here, a shift σ chosen near zero, would result in a very rapid convergence to the eigenvalue λ_2. This approach cannot be taken here, since it requires the factorization of the matrix $Q - \sigma I$, which is a large sparse symmetric matrix with the same sparsity structure as M. Our original goal, however, is to find an efficient reordering of M, so to be able to factor it efficiently. Hence the 'shift and invert' approach would require us to factor a matrix closely related to M, and thus cannot be considered in this application.

Reorthogonalization methods are a second set of techniques, which have been applied to the Lanczos algorithm, in order to improve both its reliability and computational efficiency [37, 43, 36]. For the computation of the second eigenvalue and vector, only a limited amount of reorthogonalization is necessary. No reorthogonalizations are performed at the right end of the spectrum, with respect to the large eigenvalues, since there is no interest in the accurate computation of eigenvalues at this end. Also it is unlikely that preserving orthogonality at the right end will have any impact on the convergence of the Lanczos algorithm towards the second smallest eigenvalue which is at the left end of the spectrum.

However, the first eigenvector $\underline{x}_1$ of Q is $\underline{e}$, the vector of all ones, and this vector can be used for reorthogonalization at the left end of the spectrum. At each step we explicitly orthogonalize the current Lanczos vector against $\underline{e}$. This is effectively a deflation of the problem and now the eigenpair $\lambda_2, \underline{x}_2$ will be computed as the first eigenpair on the left end of the spectrum.

Another important consideration for the Lanczos algorithm is the choice of a starting vector. In the absence of any other information, a random starting vector is appropriate. However, many practical matrix problems are presented already in an ordering relevant to the formulation of the problem, sometimes even in an ordering which is close to a good band or envelope ordering. In this case it is desirable to transmit this ordering information to the Lanczos algorithm. This was accomplished by setting the starting vector in the Lanczos algorithm to $\underline{r}$, with $r_i = i - (n+1)/2$. In this way the starting vector looked very much like a Fiedler vector for the original ordering of the matrix as presented to the algorithm. In most cases this turned out to be a better choice than a random starting vector, because it resulted in faster convergence to the second eigenvector.

Finally, another point needs to mentioned. Considering the very simple structure of the Laplacian matrix Q, and the seeming simplicity of the task of computing just one eigenpair at the left end of the spectrum, one might be inclined to avoid the complexities of the Lanczos algorithm, and attempt to solve this problem with a simple shifted power method, and a deflation procedure analogous to the one described above. This was tried as a first attempt at the computation of a second eigenvector, but with very poor results. The power method converged exceedingly slowly, in many cases exhibiting the phenomenon of *misconvergence* [38]. This meant that the power method settled down at an eigenvalue of Q, which was not the Fiedler value, and whose eigenvector correspondingly delivered a very poor reordering. The results here prove the claims of [38] to be correct that even in the simplest cases the Lanczos algorithm is the method of choice, when computing eigenvalues of large, sparse, symmetric matrices.

The above discussion can now be summarized in the following more complete description of the special form of the Lanczos algorithm:

ALGORITHM 1.

1. *Given the sparsity structure of a matrix M, form the Laplacian matrix Q.*
2. *Pick a starting vector $\underline{r}$, with $r_i = i - (n+1)/2$.*
3. *Carry out a Lanczos iteration with the matrix Q and starting vector $\underline{r}$. At each step orthogonalize the Lanczos vector against the vector $\underline{e}$. Stop when a second eigenvector has been determined to sufficient accuracy.*

In the algorithm described above, we have assumed that the Laplacian $Q(G)$ is irreducible, or equivalently that the graph G is connected. Many of the sparse matrices from the Boeing-Harwell collection have disconnected adjacency graphs. We can apply the above algorithm to each connected component separately, but there is a better approach, which we now describe.

If a graph G has k connected components, then its Laplacian $Q(G)$ has its first k eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$ equal to zero. For each connected component C_j of G, let $\underline{x}_j$ be the vector with ones in components corresponding to the vertices in C_j, and zeros in all other components. The vectors $\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_k$ can be chosen to be independent eigenvectors corresponding to the multiple eigenvalue zero. The next smallest eigenvalue λ_{k+1} is positive, and a corresponding eigenvector $\underline{x}_{k+1}$ is the vector used to partition the graph.

At each step of the Lanczos algorithm, we can now orthogonalize the Lanczos vector *simultaneously* against the first k eigenvectors $\underline{x}_1, \ldots, \underline{x}_k$, since these vectors have nonzeros in disjoint rows. This approach has the advantage that it increases the vector length in the Lanczos algorithm, and thus permits efficient vectorization.

7.2. Convergence and quality of separators. We now present our results on the number of iterations and the time required by the Lanczos algorithm as the second eigenvector is computed to a set of different tolerances. The tolerance criterion, *tol*, is the 2-norm of the residual vector $Q\underline{u} - \lambda\underline{u}$, where λ, $\underline{u}$ are the computed quantities at the current step in the algorithm. We also study the quality of the vertex separators obtained from these approximate eigenvectors.

We report results for a few representative problems from the Boeing-Harwell collection and for two grid problems in Table 5. The number of iterations reported are multiples of twelve, since we checked for convergence in the Lanczos algorithm by an eigendecomposition of the tridiagonal matrix only after every twelve iterations. Times are in seconds on a Cray Y-MP, using our vectorized Lanczos code. For each value of *tol*, we report the size of the vertex separator and the corresponding part sizes computed by the *Spectral Partitioning Algorithm*. Blank entries in the separator columns mean that the separator computed is the same as the one obtained with the previous tolerance.

For most of the problems that we have computational results, it is only necessary to compute the second eigenvector to a tolerance of about 10^{-2}, to obtain the best separator obtained by the spectral algorithm. This accuracy only requires a modest number of Lanczos iterations, and can be obtained reasonably fast. One class of notable exceptions is the power network problems, illustrated by BCSPWR10 in the table. In these problems, a vertex

TABLE 5
Convergence Results. Times are in seconds on a Cray Y-MP.

Key	Tol	Iterns	Time	$\|S\|$	$\|A\|$	$\|B\|$
NASA4704	10^{-1}	24	0.27	172	2266	2266
	10^{-2}	60	0.65			
	10^{-3}	72	0.80			
	10^{-4}	96	1.10			
	10^{-5}	108	1.30			
	10^{-6}	120	1.50			
BCSSTK13	10^{-1}	36	0.23	236	905	862
	10^{-2}	36	0.23			
	10^{-3}	48	0.30			
	10^{-4}	60	0.39			
	10^{-5}	72	0.49			
	10^{-6}	84	0.60			
BCSPWR10	10^{-1}	24	0.24	171	2619	2510
	10^{-2}	84	0.92	72	2642	2586
	10^{-3}	252	7.20	34	2643	2623
	10^{-4}	300	11.90	31	2646	2623
GRD61.101.5	10^{-2}	12	0.15	101	3050	3010
	10^{-3}	36	0.42	61	3050	3050
	10^{-4}	96	1.26			
	10^{-5}	108	1.47			
	10^{-6}	120	1.67			
GRD61.101.9	10^{-1}	12	0.16	101	3030	3030
	10^{-2}	24	0.30	61	3050	3050
	10^{-3}	108	1.53			
	10^{-4}	120	1.76			
	10^{-5}	144	2.38			
	10^{-6}	156	2.70			

has a small average degree, and the graphs have large diameters, and hence computing global information in such graphs is relatively slow. Hence a large number of iterations are necessary to compute the second eigenvector accurately. In the BCSPWR10 problem, after 300 iterations, the norm of the residual in the eigenvalue equation was about 10^{-4}. In this problem, the vertex separator decreases in size as the eigenvector becomes more accurate.

8. Conclusions. We have considered an algebraic approach to computing vertex separators, and have shown that the eigenvalues of the Laplacian matrix can be used to obtain lower bounds on the sizes of the separators. We have described a heuristic algorithm for computing vertex separators from the second eigenvector of the Laplacian. It is enough to compute the eigenvector to low accuracy to obtain good separators for most problems.

Other approaches to computing vertex separators in sparse matrix algorithms have been considered by several authors. Automatic Nested Dissection (AND) is used in SPARSPAK to compute fill-reducing orderings of sparse matrices [20]. More recently, a parallel Kernighan-Lin algorithm was employed to compute orderings appropriate for parallel factorization by Gilbert and Zmijewski [22]. The separators computed by the spectral algorithm compare favorably with those obtained by AND or the Kernighan-Lin algorithm. The results with the *Spectral Partitioning Algorithm*are not quite as good as those obtained by Leiserson and Lewis [29] and Liu [31].

The spectral algorithm has an advantage over all these algorithms in that its dominant computation is an eigenvector computation, and is fairly straight forward to compute efficiently on medium size multiprocessors used in scientific computing. For these other algorithms, it is either not clear how to implement them in parallel or the degree of parallelism is not high.

The computation of good separators is useful in any algorithm that employs the divide and conquer paradigm. In particular the *Spectral Partitioning Algorithm*will be also useful for VLSI layout problems [5]. But our immediate intent was to use the spectral separator algorithm to compute good orderings for parallel sparse factorizations. In order to accomplish our goals in this respect more work remains to be done. We intend to compute and study the quality of orderings obtained by the recursive application of the *Spectral Partitioning Algorithm.* A faster computation of the Fiedler vector using Davidson's method might be possible. Finally much remains to be understood about the theoretical underpinneings of the *Spectral Partitioning Algorithm.* How do Laplacian eigenvectors partition graphs? It will be useful to obtain results on the quality of the partitions computed by Laplacian eigenvector components. It will also be helpful to identify classes of graphs that are partitioned optimally by spectral algorithms.

Even though much more needs to be done, in order to demonstrate the usefulness of the *Spectral Partitioning Algorithm*for reordering sparse matrices in parallel our results are very encouraging for the following reasons: These results are very encouraging for the following reasons:

1. This a completely new type of algorithm for the graph partitioning problem and hence potentially for reordering sparse matrices.
2. Since the algorithm involves mainly floating point computation it becomes very competitive on very large problems on machines like the Cray Y-MP, where floating

point arithmetic is considerably faster than integer arithmetic.

3. Since the algorithm involves mainly dense and sparse vector operations, it can be easily implemented in parallel.

REFERENCES

[1] N. Alon, Z. Galil, and V. D. Milman, *Better expanders and superconcentrators*, J. Algorithms, 8 (1987), pp. 337–347.

[2] N. Alon and V. D. Milman, λ_1, *isoperimetric inequalities for graphs, and superconcentrators*, J. Comb. Theory, Series B, 38 (1985), pp. 73–88.

[3] E. R. Barnes, *An algorithm for partitioning the nodes of a graph*, SIAM J. Alg. Disc. Meth., 3 (1982), pp. 541–550.

[4] E. R. Barnes and A. J. Hoffman, *Partitioning, spectra, and linear programming*, in Progress in Combinatorial Optimization, W. E. Pulleyblank, ed., Academic Press, 1984, pp. 13–25.

[5] S. N. Bhatt and F. T. Leighton, *A framework for solving VLSI graph layout problems*, J. of Comp. System Sciences, 28 (1984), pp. 300 – 343.

[6] F. Bien, *Constructions of telephone networks by group representations*, Notices of the Amer. Math. Soc., 36 (1989).

[7] N. L. Biggs, *Algebraic Graph Theory*, Cambridge University Press, Cambridge, 1974.

[8] M. Blum, R. W. Floyd, V. R. Pratt, R. L. Rivest, and R. E. Tarjan, *Time bounds for selection*, J. Comput. & Systems Sci., 7 (1972), pp. 448–461.

[9] R. B. Boppana, *Eigenvalues and graph bisection: an average case analysis*, in 28th Annual Symp. Found. Comp. Sci, 1987, pp. 280–285.

[10] P. Buser, *On the bipartition of graphs*, Disc. Appl. Math., 9 (1984), pp. 105–109.

[11] D. M. Cvetkovic, M. Doob, and H. Sachs, *Spectra of Graphs*, Academic Press, 1980.

[12] W. E. Donath and A. J. Hoffman, *Lower bounds for the partitioning of graphs*, IBM Journal of Research and Development, 17 (1973), pp. 420–425.

[13] I. S. Duff, R. G. Grimes, and J. G. Lewis, *Sparse matrix test problems*, ACM TOMS, 15 (1989), pp. 1 – 14.

[14] I. S. Duff and T. Wiberg, *Implementations of $o(n^{1/2}\tau)$ assignment algorithms*, ACM Trans. on Math. Software, 4 (1988), pp. 267–287.

[15] T. Ericsson and A. Ruhe, *The spectral transformation Lanczos method*, Math. Comp., 34 (1980), pp. 1251–1268.

[16] C. Fiduccia and R. Mattheyses, *A linear time heuristic for improving network partitions*, in Design Automation Conference, 1982.

[17] M. Fiedler, *Algebraic connectivity of graphs*, Czech. Math. J., 23 (1973), pp. 298–305.

[18] ——, *A property of eigenvectors of non-negative symmetric matrices and its application to graph theory*, Czech. Math. J., 25 (1975), pp. 619–633.

[19] ——, *Special matrices and their applications in numerical mathematics*, Martinus Nijhoff Publishers, Dodrecht, The Netherlands, 1986.

[20] J. A. George and J. W. Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice Hall, 1981.

[21] J. A. George and E. Ng, *On row and column orderings for sparse least squares problems*, SIAM J. Numer. Anal., 20 (1983), pp. 326–344.

[22] J. R. Gilbert and E. Zmijewski, *A parallel graph partitioning algorithm for a message passing multiprocessor*, in Proceedings of the Intern. Conf. Supercomput., Springer Verlag, 1987.

[23] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 1983.

[24] R. Grimes, J. Lewis, and H. Simon, *Eigenvalue problems and algorithms in structural engineering*, in Large Scale Eigenvalue Problems, J. Cullum and R. Willoughby, eds., North-Holland, 1986, pp. 81–93.

[25] ——, *The Implementation of block Lanczos algorithm with reorthogonalization methods*, Tech. Rep. ETA-TR-91, Boeing Computer Services, Seattle, WA, 1988.

[26] A. J. HOFFMAN AND H. W. WIELANDT, *The variation of the spectrum of a normal matrix*, Duke Math. J., 20 (1953), pp. 37–39.

[27] J. E. HOPCROFT AND R. M. KARP, *An $n^{2.5}$ algorithm for maximum matchings in bipartite graphs*, SIAM J. Comput., 2 (1973), pp. 225–231.

[28] B. W. KERNIGHAN AND S. LIN, *An efficient heuristic procedure for partitioning graphs*, The Bell System Technical J., 49 (1970), pp. 291–307.

[29] C. E. LEISERSON AND J. G. LEWIS, *Orderings for parallel sparse symmetric factorization*, 1987, Third SIAM Conference on Parallel Processing for Scientific Computing.

[30] J. W. LIU, *The minimum degree ordering with constraints*, Tech. Rep. CS-88-02, Computer Science, York University, North York, Ontario, Canada, 1988.

[31] ——, *A graph partitioning algorithm by node separators*, 1989, (To appear in ACM Trans. Math. Softw.).

[32] B. MOHAR, *Eigenvalues, diameter, and mean distance in graphs*, April 1988, Preprint Series Dept. Math. no. 259, University E. K. of Ljubljana, Jadranska 19, 61111 Ljubljana, Yugoslavia.

[33] ——, *Isoperimetric numbers of graphs*, 1988, (To appear in J. Comb. Theory, Series B).

[34] B. MOHAR, *The laplacian spectrum of graphs*, in Sixth Intern. Conf. on Theory and Applic. of Graphs, Kalamazoo, Michigan, 1988.

[35] B. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, New Jersey, 1980.

[36] B. PARLETT, B. NOUR-OMID, AND Z. LIU, *How to Maintain Semi-orthogonality Among Lanczos Vectors*, Tech. Rep. PAM-420, Center for Pure and Applied Math., University of California, Berkeley, CA, 1988.

[37] B. PARLETT AND D. SCOTT, *The Lanczos algorithm with selective orthogonalization*, Math. Comp., 33 (1979), pp. 217–238.

[38] B. PARLETT, H. SIMON, AND L. STRINGER, *Estimating the largest eigenvalue with the Lanczos algorithm*, Math. Comp., 38 (1982), pp. 153–165.

[39] A. POTHEN AND C. FAN, *Computing the block triangular form of a sparse matrix*, Tech. Rep. CS-88-51, Computer Science, Penn State, Dec 1988. To appear in ACM Trans. Math. Softw.

[40] D. POWERS, *Structure of a matrix according to its second eigenvector*, in Current Trends in Matrix Theory, F. Uhlig and R. Grone, eds., Elsevier, 1987, pp. 261–266.

[41] ——, *Graph partitioning by eigenvectors*, Lin. Alg. Appl., 101 (1988), pp. 121–133.

[42] D. SCOTT, *Block Lanczos Software for Symmetric Eigenvalue Problems*, Tech. Rep. ORNL/CSD 48, Oak Ridge National Lab, Oak Ridge, TN, 1979.

[43] H. SIMON, *The Lanczos algorithm with partial reorthogonalization*, Math. Comp., 42 (1984), pp. 115–136.

[44] J. WILKINSON, *Modern error analysis*, SIAM Rev., 13 (1971), pp. 548–568.